Домашнее задание 9. Шилова Ксения
Отчет по ошибкам (или отсутствию ошибок) библиотеки javaplex

Библиотека предназначена для работы с графами и подсчету их топологических характеристик. Я взяла эту библиотеку, потому что моя работа в лаборатории на ФКН и курсовая работа связаны с гомологиями на графах. Эта библиотека (javaplex) одна из самых известных библиотек для работы с гомологиями. Кроме того, она не очень хорошо отлажена, и там вполне могут быть ошибки.

Прикладываю ссылку на репозиторий: https://github.com/appliedtopology/javaplex

1. Сначала я решила протестировать метод генерации графа (**FuzzerGraphGenerate**.java)
   Очень быстро нашлась ошибка:
```
== Java Exception: java.lang.NegativeArraySizeException: -2147483648
        at edu.stanford.math.plex4.graph.random.BinaryHierarchicalGraph.getProbabilities(Unknown Source)
        at edu.stanford.math.plex4.graph.random.BinaryHierarchicalGraph.construct(Unknown Source)
        at edu.stanford.math.plex4.graph.random.BinaryHierarchicalGraph.generate(Unknown Source)
        at com.company.FuzzerGraphGenerate.fuzzerTestOneInput(FuzzerGraphGenerate.java:12)
```
   Оказывается, там не предусмотрен случай неверных размеров массива (то есть на вход подается два числа int и double) – размер графа и вероятность построение ребра графа.

2. Следующим я тестировала метод EstimateDiameter и создание Евклидового пространства от массива точек (**FuzzerCreateMetricSpace**.java).
   Программа проработала 2 часа и ошибок не обнаружилось. Кажется, их там нет. Мне показалось, что не нужно использовать seeds, потому что каждый раз для генерации точки я использую consumeDouble метод.

3. Затем я тестировала с помощью target class = **FuzzerVietorisRipsComplex**.java, и программа спустя 1-1.5 часов работы упала с OutOfMemoryException. Прикладываю саму ошибку:
```
== Java Exception: com.code_intelligence.jazzer.api.FuzzerSecurityIssueLow: Out of memory (use '-Xmx1620m' to re

Caused by: java.lang.OutOfMemoryError: GC overhead limit exceeded
        at gnu.trove.TIntHashSet.iterator(TIntHashSet.java:135)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.addCofaces(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.incrementalExpansion(Unknown Source)
        at edu.stanford.math.plex4.streams.impl.FlagComplexStream.constructComplex(Unknown Source)
        at edu.stanford.math.plex4.streams.interfaces.PrimitiveStream.finalizeStream(Unknown Source)
        at edu.stanford.math.plex4.api.FilteredStreamInterface.createPlex4VietorisRipsStream(Unknown Source)
        at edu.stanford.math.plex4.api.Plex4.createVietorisRipsStream(Unknown Source)
        at com.company.FuzzerVietorisRipsComplex.fuzzerTestOneInput(FuzzerVietorisRipsComplex.java:16)
```